UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

Luis Miguel Pereira Mendes

PERVASIVE DATA INTELLIGENCE IN REAL TIME

Made by the orientation of:

Professor José M. Barcelo Ordinas

Professor Carlos Filipe de Silva Portela

Professor Joan Sardá

April 2018

Acknowledgments

I would like to thank all the support received by those who trust me to face this challenge.

A special thanks to my thesis advisor, professor Filipe Portela, that had shown totally available to answer my issues with clarity. Also, I want to thank the co-advisor, professor Manuel Filipe Santos.

I want to thank my thesis advisor at FIB, professor Jose M. Barcelo Ordinas that always clarify my questions regarding the rules of TFG, helping me with a new school environment and solving setbacks always with a big willing to help.

I want to thank my GEP advisor at FIB, professor Joan Sardá, that always was available to clarify question regarding the project management.

A very special mention to my parents that always supported me and believe me. Their psychological and financial support were essential and without it, this experience abroad would be even more a difficult challenge.

Abstract

This project topic had as proponent organization a web development start-up, the *IOTech,* which is a recently created company willing the best web solutions for several markets. One of them regards to a well-known topic among software companies: data. Tons of data are generated nowadays by millions of users on the web. The goal is to take advantage of this data to forecast changing trends inside companies and lead to quickly and precise market responses.

Assuming this, it was proposed to develop a prototype that gather 3 important concepts: data visual reporting, real-time visualization and a pervasive software. The prototype presented down below allow a user to create customized OLAP cubes through data stored in a NoSQL database and present dashboards and indicators.

In this document is described all the steps needed to understand how this project was done.

Key-words: Pervasive, data intelligence, business intelligence, data visual reporting, real time, OLAP, charts

Conteúdo

Acknowledgments 3
Abstract 4
Acronyms 8
1.General Objective 9
2.Context 9
2.1 Introduction 9
2.2. Academic context and Study Plan 9
2.3.Motivation 10
2.4. Expected Results and Contribution 10
2.5. Survey of existing solutions 10
3.Budget and Sustainability 11
3.1 Budget 11
3.2. Sustainability 11
3.2.1 Economical and Environmental 11
3.2.2. Social 11
3.Planning 12
3.1 Gantt Chart 12
3.State of art 14
3.1 Strategy of research: 14
3.1.2 Sources 14
3.1.3 Selection criteria 15
3.2 Pervasive Concept 15
3.3. Real-Time Concept 15
3.4. Data Visualization and Decision Support 15
3.5. OLAP and NoSQL 16
4.Methodology and Rigor 20
4.1. Problem and motivation 22
5. Development of a prototype 22
5.1 Work tools and self-learning 22
5.2.1.Server-Side 23
5.2.2.Libraries 23
5.3. Coding Architecture 27
6.Results 28
6.1. Data Visualization 28
6.2. OLAP creation 29

6.3.Real-time 32
7.Conclusions 33
7.1. Objectives 33
7.2. Personal conclusion 33
7.3. Future work 33
References 34

Figura 1 Gantt Chart overview 12
Figura 2 - Gant Chartt – GEP 12
Figura 3 Gantt Chartt – Research 13
Figura 4 Gantt Chart - Requirement Analysis and Design 13
Figura 5 Gantt Chart – Development 13
Figura 6 Gantt Chart - TFG Delivery 14
Figura 7Google Analytics example - Taken from Google Analytics webpage 16
Figura 8 Data Warehousing - Taken from An Overview of Data Warehousing and OLAPTechnology (Chaudhuri, Surajit et al.) 17
Figura 9 Multidimensional data - Taken from An Overview of Data Warehousing and OLAPTechnology (Chaudhuri, Surajit et al.) 17
Figura 10 JSON Document 18
Figura 11 Collection of documents 18
Figura 12 Example of document inside a collection 20
Figura 13 - Design Science Research Process Model (DSR Cycle) – Taken from A Design Science Research Methodology for Information Systems (Ken Peffers et al., 2007) 21
Figura 14- Requiring express module 23
Figura 15- Server config 23
Figura 16- Middleware for views 23
Figura 17 Boddy parser middleware 24
Figura 18 Set static folder 24
Figura 19 Enable CORS 24
Figura 20 Middleware for express session 24
Figura 21 Passport Initialization 24
Figura 22 Express validator middleware 25
Figura 23 Connect flash middleware 25
Figura 24https://www.npmjs.com/package/tiny-olap 26
Figura 25Result of Tiny-Olap implementation 26

Acronyms

DB	Database
DSR	Design Science Research
ECTS	European Credit Transfer and Accumulation System
ES6	ECMAScript 6
GEP	Gestió de Projectes
JS	JavaScript
JSON	JavaScript Object Notatio
NoSQL	Not Only SQL
NPM	Node Package Manager
OLAP	Online Analytical Processing
TFG	Treball de Fi de Grau
UI	User Interface
URL	Uniform Resource Locator

1.General Objective

The main objective of the TFG is develop a web solution that allow users to make data science analysis in real time.

The focus of the solution is Business Intelligence as a Service. It allows to fetch a dataset from a database and visual report it. Also, this solution should be pervasive, that is, should be possible to access it anywhere at anytime through a device with internet connection.

2.Context

2.1 Introduction

With the growth of the amount of data on the web, data analysis has taken on a more important role than ever. Generated through computers, mobile devices, notebooks, etc, data is everywhere and it allows each day more and more precise behavioral insights and forecasting in the every field. Being left behind , leading to best decisions, is what every company wills to.

Real-time decision support based on currently generated data is seen by organizations as a decisive factor for success in taking decisions. However, organizations have difficulty in analyzing these data in real time due to their complexity, quantity and diversity. Based on this assumption it is expected the development of a prototype that facilitates the process of Data Analytics, i.e., the analysis of data in Real-time, allowing them to be accessed / analyzed anywhere and at any time.

2.2. Academic context and Study Plan

The author of this project is an Erasmus student from the University of Minho, Portugal. Currently in the 5th year of the course “Integrated Master's in Engineering and Management of Information System” in UMinho, the student is taking the pre-dissertation at UMinho (15 ECTS) as equivalence to the TFG- *Treball de Fi de Grau* (18 ECTS) at UPC.

After concluding this stage at UPC, the student should finish the dissertation at University of Minho, with 15 for the pre-dissertation and 30 ECTS for the dissertation. Given this situation, the thesis advisor at UMinho, Filipe Portela, proposed to divide the final software in two steps:

- Complete the app logic at UPC

- Update the prototype with a front-end JS framework + API documentation

In this report, it's abstracted the second step since it will be done after this stage.

2.3.Motivation

It's common knowledge that data is increasing very fast nowadays and consequently the possibilities to make consumers predictions are getting an important paper in prediction making. The way that companies look to data visualization softwares is now seen as a must have. The demand for this softwares is higher than ever and the chance to contribute to these areas is very gratifying.

2.4. Expected Results and Contribution

It is expected that this project covers the following points:

- Survey of existing solutions;
- Prototype of a web solution with real-time features;
- Prototype of a web solution with data visualization features;
- Development of an interface with pervasive features;
- Validation of the prototype;
- New knowledge that combines the area of information systems, web programming and data science;
- Improvement in the real-time data analysis process;

2.5. Survey of existing solutions

Currently the market offers few solutions for data visualization. Down below are listed the most relevant:

-Dundas;

-ClicData;

-Pentaho;

-icCube Data Analysis & Reporting Software;

The list of OLAP tools goes on. Although these solutions provide several features regarding data visualization, the solution created here is a free and lightweight browser based software that allow everyone to make data visualization.

3.Budget and Sustainability

3.1 Budget

This project doesn't represent any direct cost. The softwares used throughout the development of the project are free. The only indirect cost that present is the portable computer which can be calculated by the follow calculation:

Toshiba Portege Z930 Estimated lifetime = 10 years

Price = 899€

Project duration = 156 days = 0,42 years

Depreciation = (0,42 years * 899€) / 10 years = 37,76 €

Total costs: 37,76€.

3.2. Sustainability

3.2.1 Economical and Environmental

Environmental costs in this project will be insignificant since the only resource needed during the project will be a computer with internet connection. The electricity will lead to costs and emissions of CO2 which we calculate below.

Average electricity source emissions of CO2: 1.222lbs CO2 per kWh

Average price of electricity in Spain: 0.12 €/kWh

Average laptop consumption: 60 watts

Total hours: 1856h

Total kW: 60w * 1856h = 111 360w = 111,360 kW

Total price: 111,360kWh * 0,12€ = 13, 36 €

Total CO2: 111,360kWh * 1.222lbs = 136,08 lbs C02 = 61,72 Kg

3.2.2. Social

The solution will be free and accessible for everyone with a device with internet connection. It pretends to help the client/user to get better insights of his company through analysis in real time. As well as the client will benefit of this new solution, the author will have opportunity to increase his knowledge about such technologies and skills.

3.Planning

3.1 Gantt Chart

Gantt chart is an important tool to schedule task and make sure they are workable.

During this project the author faced unpredictable time consumptions such as other very time consumption courses.

Given that, the author decided to postpone the final delivery to the extraordinary period.

Task Name	Duration	Start	Finish
TFG	156 days	Mon 18/09/17	Mon 23/04/18
Phase 1 - GEP	35 days	Mon 18/09/17	Fri 03/11/17
Phase 2 - Research	52 days	Fri 03/11/17	Mon 15/01/18
Phase 3 - Requirements Analysis and Design	13 days	Mon 15/01/18	Wed 31/01/18
Phase 4 - Development	50 days	Fri 02/02/18	Thu 12/04/18
Phase 5 - TFG Delivery	7 days	Fri 13/04/18	Mon 23/04/18

Figura 1 Gantt Chart overview

There are 5 main phases regarding this project. They are the project management(GEP), Research, Requirements Analysis and Design, Development and TFG Delivery.

Phase 1 – GEP

Phase 1 - GEP	35 days	Mon 18/09/17	Fri 03/11/17
Deliverable 1	6 days	Mon 18/09/17	Mon 25/09/17
Deliverable 2	6 days	Mon 25/09/17	Mon 02/10/17
Deliverable 3	6 days	Mon 02/10/17	Mon 09/10/17
Deliverable 4	6 days	Mon 09/10/17	Mon 16/10/17
Deliverable 5	6 days	Mon 16/10/17	Mon 23/10/17
Deliverable 6	6 days	Mon 23/10/17	Mon 30/10/17
Oral defence	5 days	Mon 30/10/17	Fri 03/11/17

Figura 2 - Gant Chartt – GEP

The project management is divided by 6 deliveries and an oral defence. Includes the context and scope of the project, project planning, budget and sustainability, first oral presentation, oral presentation and final document.

Phase 2 - Research	**52 days**	**Fri 03/11/17**	**Mon 15/01/18**
Awareness of the problem	6 days	Fri 03/11/17	Fri 10/11/17
Define research	1 day	Fri 10/11/17	Fri 10/11/17
Research	47 days	Fri 10/11/17	Mon 15/01/18

Figura 3 Gantt Chartt – Research

Although the awareness of the problem was in part included in the contents of the GEP, the author decided to take a extra time to clarify previous ideas.

The definition how the research was going to be handled, was a important factor to access the right information with the best trustworthiness.

The research took a big part of the project. Understand concepts it's crucial before start anything. For instance, how to correlate OLAP cubes with MongoDB was a hard concept to retain. The author will continue his researches in a next stage in order to make a clear vision of these concepts and improve the artifact.

Phase 3 - Requirements Analysis and Design	**15 days**	**Mon 15/01/18**	**Fri 02/02/18**
Requirements Analysis	8 days	Mon 15/01/18	Wed 24/01/18
Design	8 days	Wed 24/01/18	Fri 02/02/18

Figura 4 Gantt Chart - Requirement Analysis and Design

Before start creating something it's important to know what should be created. The requirements analysis were done along with the thesis tutor , professor Portela, since the first given information was very wide and imprecise. Although the requirements was defined in this phase, they were changed during the development as the ideas of the author change.

Phase 4 - Development	**50 days**	**Fri 02/02/18**	**Thu 12/04/18**
Research of technoogies and tools	6 days	Fri 02/02/18	Fri 09/02/18
Self-learning	21 days	Fri 09/02/18	Fri 09/03/18
Development of a prototype	19 days	Fri 09/03/18	Wed 04/04/18
Deployment	1 day	Thu 12/04/18	Thu 12/04/18

Figura 5 Gantt Chart – Development

After have the requirements and the design, the most important phase was handled in with four sub-steps. First the author researched about the best technologies and tools to implement this solution along with his tutor. Following that, the author needed to improve his capabilities to deal with new programming languages, databases and libraries. Finally, the development and deployment. The author had several difficulties to lead with new programming procedures that had never been handled before. With the self-learning, the documentation and other examples found on IT websites, the author could overcome problems in a general way.

Phase 5 - TFG Delivery	**7 days**	**Fri 13/04/18**	**Mon 23/04/18**
Follow up report delivery	2 days	Fri 13/04/18	Mon 16/04/18
Final report delivery	1 day	Mon 16/04/18	Mon 16/04/18
Presentation slide show	5 days	Mon 16/04/18	Fri 20/04/18
TFG presentation	1 day	Mon 23/04/18	Mon 23/04/18

Figura 6 Gantt Chart - TFG Delivery

In this phase, was written a report about the whole project, covering every phase from the GEP till the Development. Here are the main ideas of the author to reach the final product.

3.State of art

3.1 Strategy of research:

The research was made mainly by search the following words:

Data intelligence, real time computing, pervasive computing, OLAP, business intelligence, node.js olap libraries.

3.1.2 Sources

Regarding the repositories for research, several were considered, namely:

- University of Minho repository

-Google Schoolar

-Scopus

-Springer Link

- Google

3.1.3 Selection criteria

The selection of the used articles was based essentially on the following characteristics: date of publication date, number of citations and finally the relation between the article and this project.

3.2 Pervasive Concept

Pervasive it's not the most popular word but it's in the other hand a simply concept to understand. In the Merriem-webmaster dictionary the word pervasive is defined as the follow "*existing in or spreading through every part of something*". In the context of our project this definition is applicable. The prototype should exist in every part, that is, should be accessible everywhere through a device provided with a browser and internet connection. That means simply host our app in a web server. The advantages are obvious for the user. For instance, he can check market trends in real-time while waiting to meet a costumer, taking this market trends into account in his decision.

3.3. Real-Time Concept

Given the context, real time is a crucial feature. Users wants actual data to forecast precisely. There is no time definition for real-time but a real time feel. The user should handle fresh data and feel that data is updated almost without latency. This can be obtained with techniques like sockets or polling. In this project, the real time feel it's obtained by using a simple function called *setInterval()* which consist in making requests in a given interval of time. Although it's not the best performance solution, following works should be made to implement *websockets* in the next phase at University of Minho.

3.4. Data Visualization and Decision Support

Data visualization is, as the word says, a representation of data for an easy way to interpret and help to take decision. To have an idea, an example could be a complex dataset being represent through a graph where the user can take from that understandable information. An example of data visualization is the Google Analytics solution.

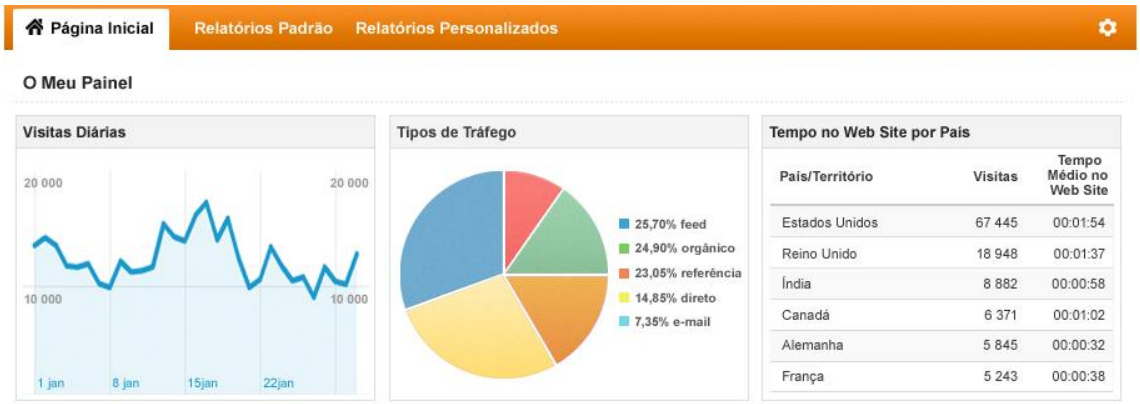

Figura 7Google Analytics example - Taken from Google Analytics webpage

Charts and reports helps users to read information and take decisions from that. The idea behind this solution is the same. Provide to users a custom visualization for a particular business case.

3.5. OLAP and NoSQL

Before going deep into the on-line analytical processing concept, it is important to clarify that this solution uses a single MongoDB database with predefined documents. Thus, Extract, Transform and Load data (ETL) is not handled in this phase by this solution. A pre-structured dataset will be used to illustrate the main functionality of this application: data visualization. Given that, the data will be directly fetched from a collection stored in a NoSQL database. This solution follow a non-relational document-oriented database, which differs from the usual relational database.

It's important to clarify that OLAP comes usually associated with data warehouses. For Chaudhuri, Surajit et al.(1997) "operational databases are finely tuned to support known OLTP workloads, trying to execute complex OLAP queries against the operational databases would result in unacceptable performance. Furthermore, decision support requires data that might be missing from the operational databases; for instance, understanding trends or making predictions requires historical data, whereas operational databases store only current data. Decision support usually requires consolidating data from many heterogeneous sources: these might include external sources such as stock market feeds, in addition to several operational databases." Given these reasons, ETL and Data warehousing are concepts that comes together with OLAP when dealing with several data sources. Data needs to be cleaned and selected carefully to answer a specific business process.

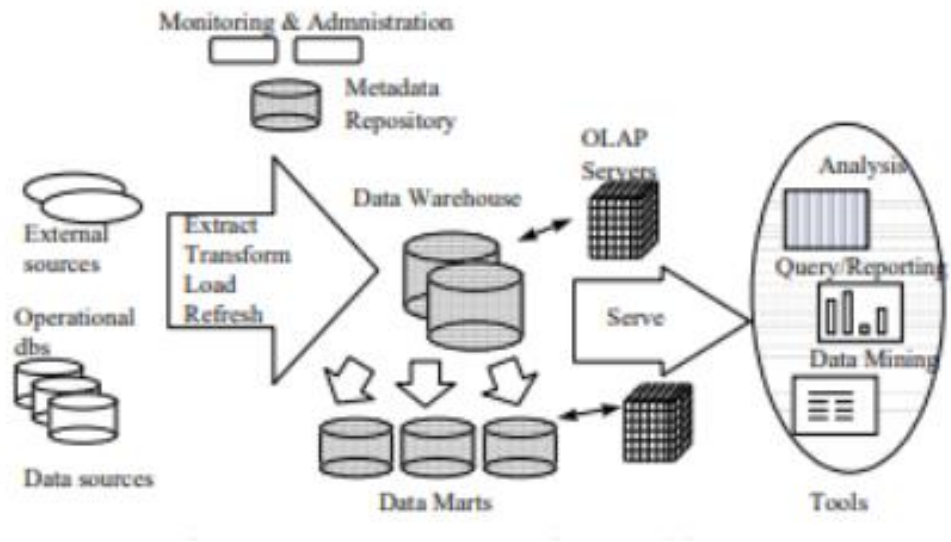

Figura 8 Data Warehousing - Taken from An Overview of Data Warehousing and OLAPTechnology (Chaudhuri, Surajit et al.)

With a single data source with predefined clean data, there is no need to ETL and create data warehouse. MongoDB will be our data source and we will create an Olap cube directly over that data.

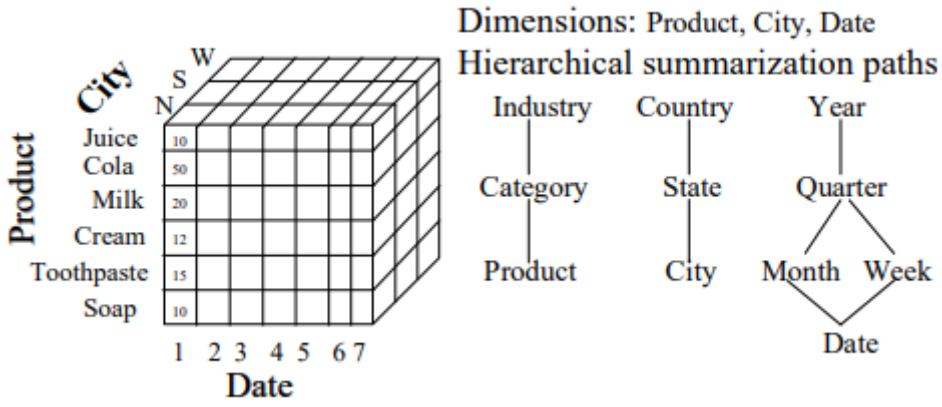

Figura 9 Multidimensional data - Taken from An Overview of Data Warehousing and OLAPTechnology (Chaudhuri, Surajit et al.)

Once the data is accessible, must be choose the dimensions and indicators to make data visualization. Above its shown an example where is represented an Olap cube with 3 dimensions (Product, City and Date) and an indicator(Quantity). This dimensions and indicators are selected in order to give to the user a sight about a specific desirable business context/process.

Knowing that, in this solution it's used the same multidimensional principles. Documents are JSON files grouped into collections. A document can contain atomic values or nested documents values:

```
var document= { //This is a document
                name: "Luis",
                age: 26,
                university: { //This is a nested document
                                name: "University of Minho",
                                course : "Information Systems"
                              }
              }
```

Figura 10 JSON Document

Whereas a collection is a set of documents:

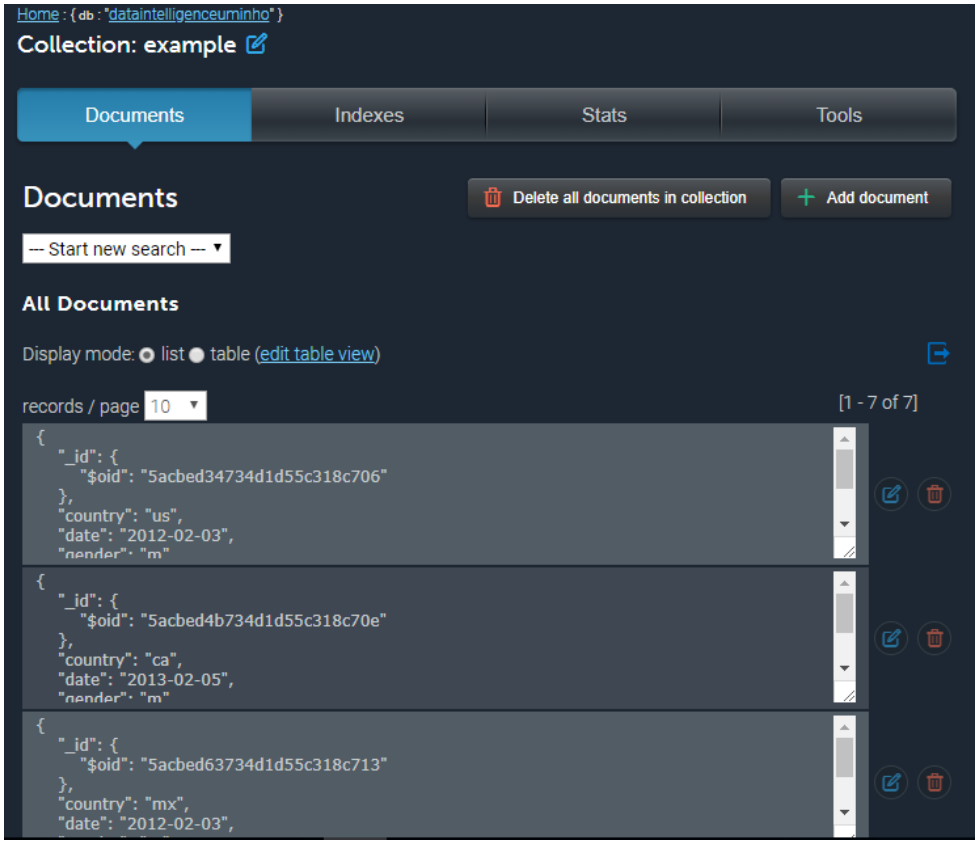

Figura 11 Collection of documents

With this in place, it's possible to reproduce the same strategy used by the relational databases to create analytical cubes, where documents provide the dimensions and indicators. For example, a collection of sales with JSON documents within keys/values for salesman, products and number of sales. Here, dimensions are the salesman and products and the indicator is the number of sales.

To create an OLAP cube we need to keep in mind the fundamental concepts defined by Ralph Kimball (2013):

“Four-Step Dimensional Design Process

The four key decisions made during the design of a dimensional model include:

1. Select the business process.
2. Declare the grain.
3. Identify the dimensions.
4. Identify the facts”

Business Process – Kimball, R(2013)-*“The operational activities performed by the organizations.”*

It’s important to have a good overview of the organization in order to select relevant business processes. A business process events generates metrics and indicators which will be used to represent the behavior of the process. For example, the business process of a product delivery will have as metric the successful deliveries or the time spent in each delivery.

Declare the grain – Kimball, R(2013)- “The grain establishes exactly what a single fact table row represents.” For example, an individual boarding pass, a daily stock level for each product or a monthly balance for each bank account.

Identify the dimensions – Kimball, R(2013)- “Dimensions provide the “who, what, where, when, why, and how” context surrounding a business process event.” It involves answering the question, "How and at what level of detail do business people describe the data that results from the business process?"

Facts – Kimball, R(2013)- “Dimensions provide the “who, what, where, when, why, and how” context surrounding a business process event.” It implies answering the question, "What are we measuring?" Candidate facts must be consistent with the grain declared in step 2. Examples: quantity or value.

After following these rules, a multidimensional structure that “consists in a fact table linked to dimensional tables via primary key/foreign key relationships” must be selected. These assumptions are taken into account with NoSQL databases in the way that Olap needs to follow rules to select dimensions and indicators.

Conversion into a NoSql documented oriented model

With NoSQL document-oriented model the data is organized in documents and these documents can contain nested documents. In this solution, only wil be used atomic values and not compound values (nested documents).

Given that, a collection will represent a database and a document will represent one or more dimensions.

One or more dimensions can be defined. As mentioned above there are usually four popular dimensions among analysts. They are:

- When?
- What?
- Who?
- Where?

After have in mind our dimensions, it's possible to step further and look for a numeric key performance indicator which usually is a numeric value.

Summarizing all this information, an example of this could be a collection with documents with the following structure:

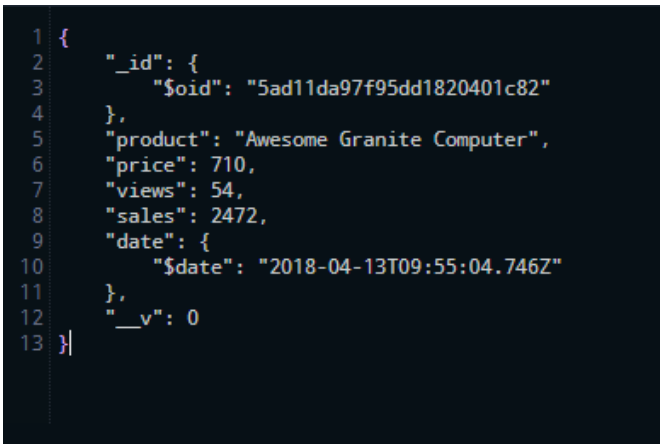

Figura 12 Example of document inside a collection

We could take product as dimension and the sales as KPI (numerical value).

4.Methodology and Rigor

For Yin (2009), the questions related to research have a crucial role. According to this author "Defining the research questions is probably the most important step to be carried out in research studies". The project analysis aims to give a clear vision about what is about to being answered.

In the case of this project, investigate the feasibility of data analysis solutions raised issues such as:

-How can customizable data analysis can be implement in NoSQL databases?

-Will this solution be capable to deal with not pre-structured datasets?

-What it will do that others don't do?

To try to answer these issues the author used the **design science research**:

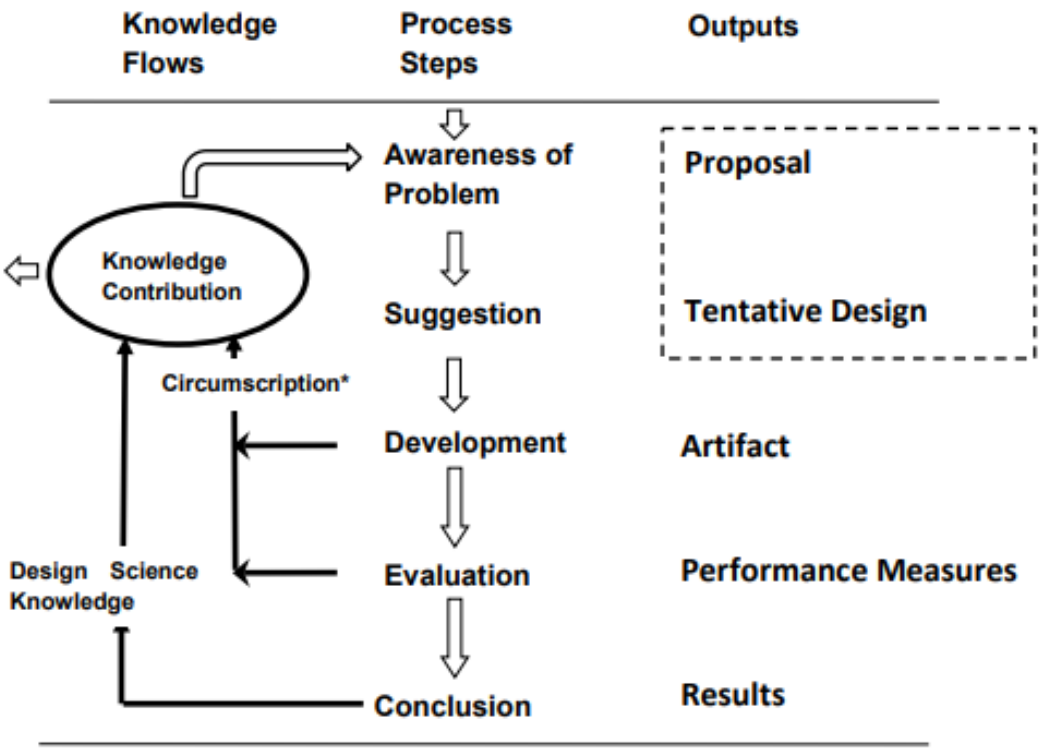

Figura 13 - Design Science Research Process Model (DSR Cycle) – Taken from A Design Science Research Methodology for Information Systems (Ken Peffers et al., 2007)

DSR is an iteration-based methodology that provided strong orientation in this project.

1. **Awareness of the problem**

   According to Vaishnavi et al. (2008) "the output of this phase is a Proposal, formal or informal, for a new research effort". In this project, the definition of the problem was given by the tutor at university of Minho. The root of the problem is the lack of alternatives to the internal software's made by companies for data science. The motivation is to create a solution that let users easily analyze data online.

2. **Suggestion**

"A Tentative Design and the performance of a prototype based on that design would be an integral part of the Proposal. Moreover, if after investing considerable effort on an interesting problem a Tentative Design or at least the germ of an idea for problem solution does not present itself to the researcher, the idea (Proposal) will be set aside."

Before getting into the development, the author made sure that through a prototype was executable. The artifact should be able to allow users to make data

science through an OLAP approach. The user should be able to forecast events based on the analysis. This analysis can be created by the own user.

3. **Development**

"The Tentative Design is further developed and implemented in this phase." With the last step in mind, it is expected to get deep into details and develop a workable solution. In this stage, the author makes use of the methodology SCRUM with sprints between 1 and 3 weeks.

4. **Evaluation**

"Once constructed, the artifact is evaluated according to criteria that are always implicit and frequently made explicit in the Proposal (Awareness of Problem phase). Deviations from expectations, both quantitative and qualitative are carefully noted and must be tentatively explained." Through experimental tests and measures, a positive or negative answer should be taken. It's common that in the first development artifact the evaluation leads to another process initiation to fulfill ideas with more research.

5. **Conclusion**

"The finale of a research effort is typically the result of satisficing, that is, though there are still deviations in the behavior of the artifact from the (multiple) revised hypothetical predictions; the results are adjudged "good enough." That is, when the artifact has a reasonable and expect behavior, it's consider good enough to conclude the research and present to an audience.

4.1. Problem and motivation

5. Development of a prototype

5.1 Work tools and self-learning

This web application was build using a single programming language: Javascript.

Javascript is a language that became more popular recently with the emergence ofNode.js , a code interpreter for JS on the server side. "Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient." – nodejs.org

Node.js applications are very fast comparing with other server languages such as PHP. The fact that uses an asynchronous mechanism (non-blocking I/O) make the usage the of the CPU more efficient since it doesn't need to wait until a request finish.

The execution of the code keeps going and the request is hold in a callback function. When the request is finish the return of the callback function is held in queue.

5.2.1.Server-Side

To develop the logic of this application the author used JavaScript running on Node.Js. The author never used this language as a server-side language. To improve and complement his knowledge, the author took online courses on Udemy, the *Complete Course of Node Developer JS and MongoDB*, instructed by Jorge Sant Ana and the *complete Node.js Developer Course (2nd Edition)*, instructed by Andrew Mead, documentation and several free online tutorials

Following that, the author searched for packages, in the NPM website, that could complement the project desired requirements.

5.2.2.Libraries

Node.js environment have a big community of developers that creates libraries with good features, practices and some of them well known among Node developers. Given that, there is no need to reinvent the wheel and create new code for frequent tasks such as login, data verification, etc. That would be unproductive. Given that, the author used the following packages:

5.2.2.1Express.js

Express.js is a JS framework provide that provide features to deal with the server. Down below, its shown how the server is configured.

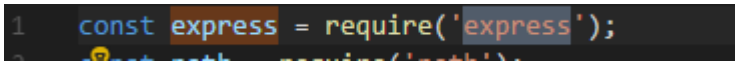

Figura 14- Requiring express module

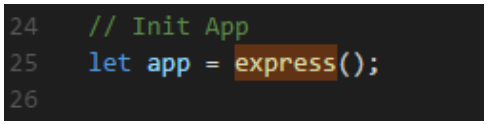

Figura 15- Server config

In the app object we set the middleware for the app:

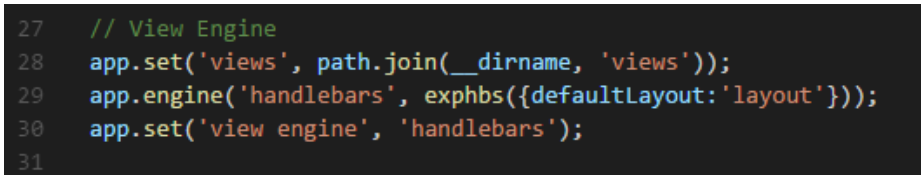

Figura 16- Middleware for views

```
// BodyParser Middleware
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: false }));
app.use(cookieParser());
```

Figura 17 Boddy parser middleware

```
// Set Static Folder
app.use(express.static(path.join(__dirname, 'public')));
```

Figura 18 Set static folder

```
//Enable CORS
app.use(cors());
```

Figura 19 Enable CORS

```
// Express Session
app.use(session({
    secret: 'secret',
    saveUninitialized: true,
    resave: true
}));
```

Figura 20 Middleware for express session

```
// Passport init
app.use(passport.initialize());
app.use(passport.session());
```

Figura 21 Passport Initialization

```
// Express Validator
app.use(expressValidator({
  errorFormatter: function(param, msg, value) {
      let namespace = param.split('.')
      , root    = namespace.shift()
      , formParam = root;

    while(namespace.length) {
      formParam += '[' + namespace.shift() + ']';
    }
    return {
      param : formParam,
      msg   : msg,
      value : value
    };
  }
}));
```

Figura 22 Express validator middleware

```
// Connect Flash
app.use(flash());
```

Figura 23 Connect flash middleware

These middleware function are used to configure our server. Description is enough to understand how the app was set. More detailed information is provided by the documentation in ExpressJS website.

5.2.2.2.Mongolab-data-api

The author of this library, Gabe Montalvo, defines it as a “a node.js module designed to allow you to access mLab's Data API with minimal overhead.”

It allows fetching collections and documents from the mLab, a cloud MongoDB service.

5.2.2.3.Tiny-Olap

Tiny-Olap is the engine responsible for the data being shown trough dashboards. Tiny-olap let creates an Olap cube with dimensions, indicators and filtering data. In this project we will create a totally custom cube through user input generating custom visualizations.

Example of implementation:

```
var TinyOlap = require('tiny-olap');
var olap = new TinyOlap(data);

var result = olap.query()
        .group(['country', 'gender'])
        .measure({name: 'pageviews', formula: 'pageviews', agg: 'sum'})
        .run();
```

Figura 24https://www.npmjs.com/package/tiny-olap

Example of a result:

```
C:\Users\Luis Mendes\Desktop\data-intelligence-miegsi>node playground.js
[ { country: 'us', gender: 'm', pageviews: 10 },
  { country: 'ca', gender: 'm', pageviews: 20 },
  { country: 'mx', gender: 'm', pageviews: 72 },
  { country: 'mx', gender: 'f', pageviews: 2 },
  { country: 'port', gender: 'm', pageviews: 6 } ]

C:\Users\Luis Mendes\Desktop\data-intelligence-miegsi>
```

Figura 25Result of Tiny-Olap implementation

5.2.2.4.Chart.js

"Chart is a simple and functional charting library which currently supports bar charts. Implementations are done on-top of a HTML5 canvas element."- in NPM chartjs library

Charts was the easiest and prettier way found to show the collected data. It provides several graph types with a very intuitive UI.

5.2.2.5. jQuery

jQuery is a JS library used to interact with HTML. Although today we can find other solutions that replace jQuery, it still be very worthy to work with. In this project, jQuery was used to make asynchronous calls using ajax.

5.2.3. Database

This application uses MongoDB, a NoSQL database. According to MongoDB website, data is stored "in flexible, JSON-like documents" which means that we can build an application without a previous defined Schema. JSON documents can change its structure over the time. With MongoDB you can store documents in the same collection with different key-value pairs.

In this app we use mLab, a Database as a Service for MongoDB. Since this an academic project we used the plan type

5.2.4. Deployment

The deployment of the solution is used via Heroku which is a plataform as a service that supports several programming languages, including the ones used in this project. The process of deployment is made running the following commands in the Command Line:

- heroku create

- git push Heroku master

5.3. Coding Architecture

It's important to keep a clean architecture while coding so the application can scale properly and be understand by other programmers. The author decided to use an MVC architecture with following folder organization.

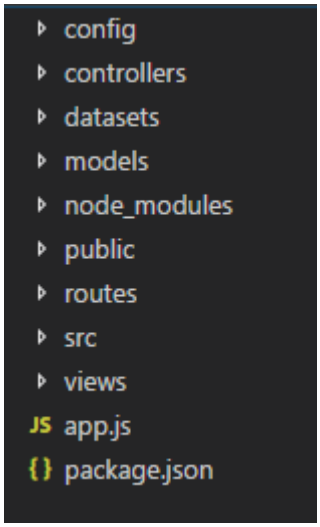

Figura 26 Architecture MVC

6.Results

6.1. Data Visualization

The main goal of this app is to help decision support. To show the data this solution uses the library Chart.JS. This library provides several types of charts. In this solution, it's possible to change between two types: bar chart and line chart.

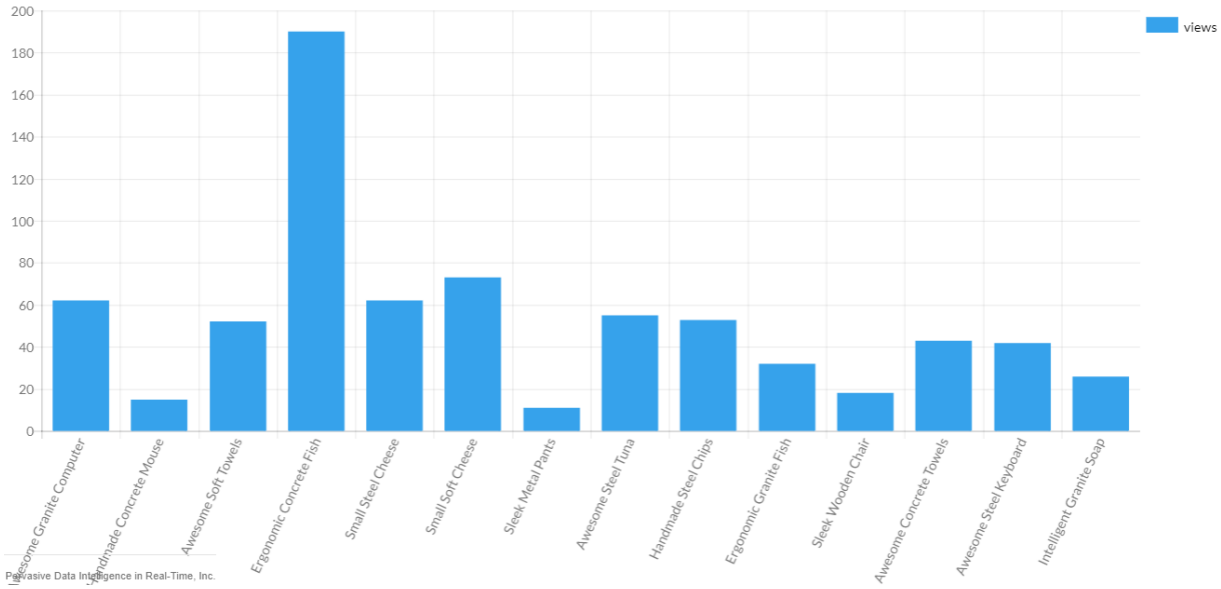

Figura 27 -Results - Bar Chart

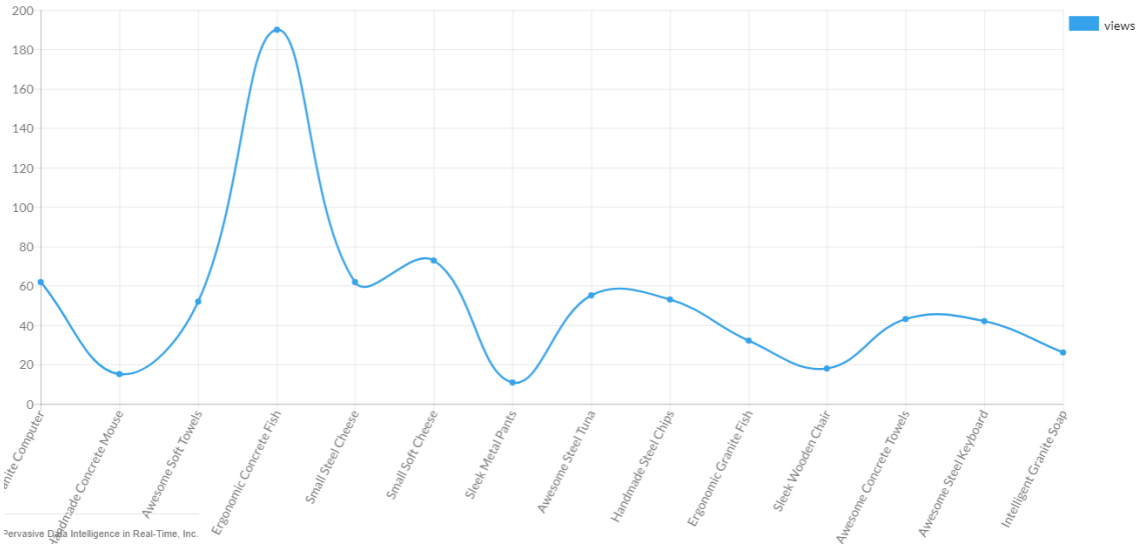

Figura 28 Results -Line Chart

These charts have no limit of data in both axis. The X axis represents the dimension and the Y axis represents the indicator.

6.2. OLAP creation

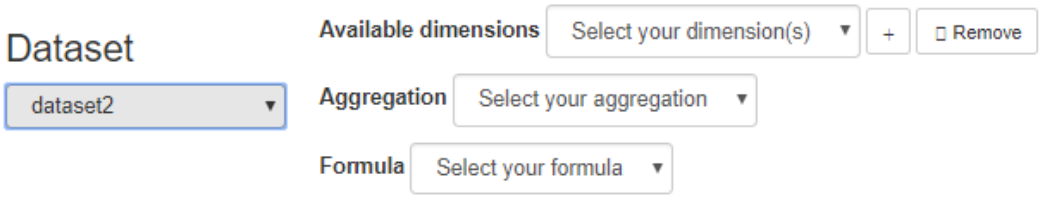

Figura 29 Result - OLAP creation

In this section, the user selects the options needed to create an Olap cube. As seen down below, these options are fetched from our DB. The submission of the option data is made on fly using jQuery.ajax() so that it's not needed to go back and forth with page refreshes like in a HTML post form.

```
164  ChartWidget.prototype.updateChart = function() {
165    var self = this;
166    if (this.formula != null && this.dimension != null && this.aggregation != null) {
167      //register data
168      let data = $.ajax({
169        url: "/dashboards/"+this.selectedDataset+"/chartData",
170        type:"GET",
171        data: {
172          formula: this.formula,
173          dimension : this.dimension,
174          agg: this.aggregation
175        },
176      }).done(function (response){
177        self.buildGraph(response);
178      })
179    }
180  }
```

Figura 30 Result- Ajax call

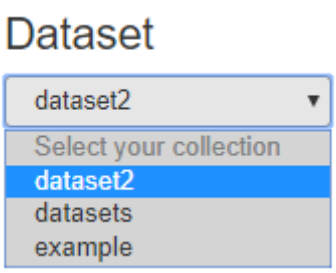

Figura 31 Result - Collection select

```
// get all the collections
mLab.listCollections('dataintelligenceuminho', function (err, collections) {
    data.collections = collections;
});
```

Figura 32 Results - Fetch all collections from a given DB

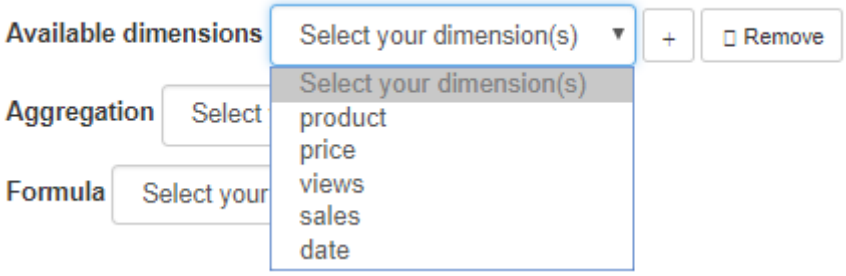

Figura 33 Result -Select dimension

```
mLab.listDocuments(options, function (err, documents) {
    //console.log(data); //=> [ { _id: 1234, ...  } ]
    let keys = Object.keys(documents[0]);
    res.render('dashboards/index',{keys:keys});
});
```

Figura 34Result - Fetch all documents from a given DB

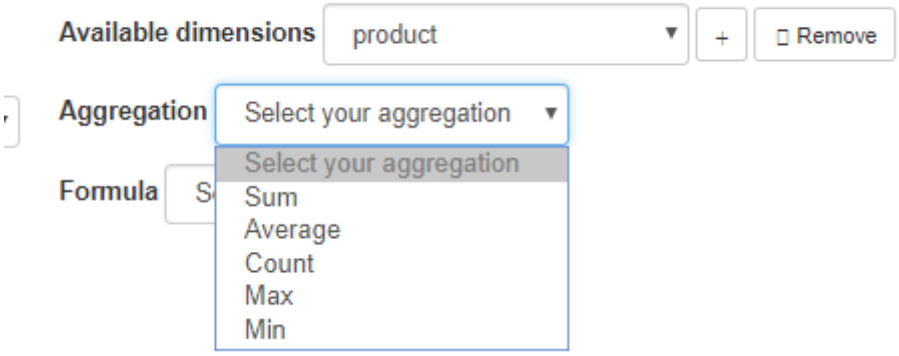

Figura 35 Result - Select aggregation

The option values are given by the library *Tiny-Olap.*

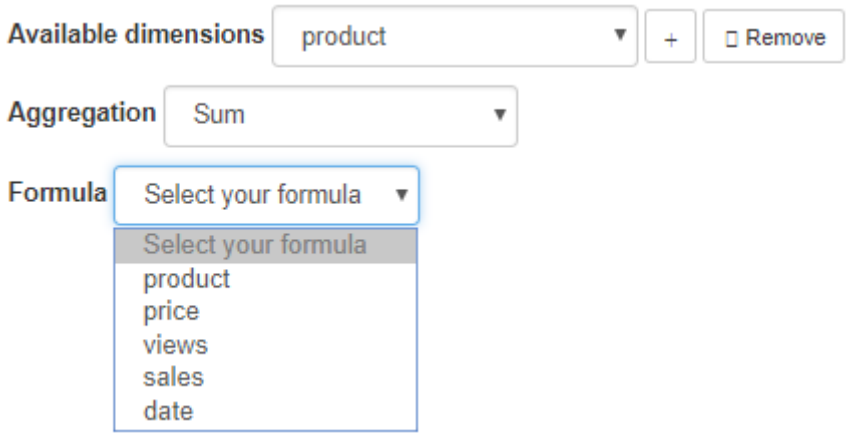

Figura 36Result Select formula

For fetching the formula, we use the same procedure as in dimensions.
mLab.listDocuments()

```
data.dataset.find({}).then(dataset => {
    let olap = new TinyOlap(dataset); //OLAP creation
    let result = olap.query().group([data.dimension1, data.dimension2]).measure({
        name: data.formula,
        formula: data.formula,
        agg: data.agg
    }).run();
    console.log(result)
```

Figura 37 Results -Olap creation

```
[ { product: 'Awesome Granite Computer', price: '710', views: 54 },
  { product: 'Handmade Concrete Mouse', price: '23', views: 15 },
  { product: 'Awesome Soft Towels', price: '1000', views: 52 },
  { product: 'Ergonomic Concrete Fish', price: '37', views: 39 },
  { product: 'Small Steel Cheese', price: '180', views: 62 },
  { product: 'Small Soft Cheese', price: '71', views: 73 },
  { product: 'Sleek Metal Pants', price: '74', views: 11 },
  { product: 'Ergonomic Concrete Fish', price: '47', views: 151 },
  { product: 'Awesome Steel Tuna', price: '47', views: 55 },
  { product: 'Handmade Steel Chips', price: '88', views: 53 },
  { product: 'Ergonomic Granite Fish', price: '16', views: 32 },
  { product: 'Awesome Steel Keyboard', price: '60', views: 42 },
  { product: 'Awesome Granite Computer', price: '450', views: 8 } ]
^CTerminate batch job (Y/N)? Y
```

Figura 38 - Results - Olap data

6.3.Real-time

```
198
199     setInterval(function(){
200         self.updateChart();
201     }, 10000);
202
```

Figura 39 Real time - setInterval()

Real time feeling is obtained via this function that makes an auto-call every 10000 milliseconds (10 seconds). Inside this function there is another function called updateChart(). Down below it's possible to see how it works:

```
158 ChartWidget.prototype.updateChart = function() {
159   var self = this;
160   if (this.formula != null && this.dimension != null && this.aggregation != null) {
161     //register data
162     let data = $.ajax({
163       url: self.CHART_DATA_ENDPOINT,
164       type:"GET",
165       data: {
166         formula: this.formula,
167         dimension : this.dimension,
168         agg: this.aggregation
169       },
170     }).done(function (response){
171       self.buildGraph(response);
172     })
173   }
174 }
175
```

Figura 40 Real time - updateChart()

The updateChart() function makes sure through an if statement if the all three values needed in the Olap creation are not null. Then, through a jQuery.ajax() call, a GET request to a predefined URL is made, where is attached a data object containing values used in the server side for the Olap creation. After that, the call back function done() retrieves the olap dataset needed to create the graph.

7.Conclusions

7.1. Objectives

The objectives previously defined were achieved in general. This solution provides data visualization, with a real-time feel and with pervasive feature.

7.2. Personal conclusion

The author of this project made a big personal improvement in the area of web development and data science. Although too much can be improved, this software can be a start point to build a powerful tool with real world usage.

7.3. Future work

Concluded the first stage of this work at UPC in Barcelona, the author should keep his work at University of Minho in order to improve the prototype. The research methodology should be re-iterated until the prototype represents a reasonable handling of the of objectives that is proposed. After that, the author should move to 2nd stage defined previously: Document an API and implement a JS front-end framework.

Although this solution is capable to give custom forecasts, it covers a small area of the world of data analytics. Other features can be implemented over this solution in order to answer wider concepts. Since this work is divided in two phases, the author, followed by the orientation of the thesis advisor, should improve these concepts.

References

Satyanarayanan, M. - Pervasive Computing: Vision and Challenges, School of Computer Science, Carnegie Mellon University (2001). http://www.cs.cmu.edu/~./aura/docdir/pcs01.pdf

Yin, R. K. (2009). Case Study Research: Design and Methods. SAGE Publications. https://books.google.com/books?id=FzawIAdilHkC&pgis=1

Inmon, W.H.(2000) What is a Data Warehouse? http://repository.binus.ac.id/2009-2/content/M0584/M058459913.pdf

Chaudhuri, Surajit; Dayal, Umeshwar(1997); An overview of Data Warehousing and OLAP Technology. Newsletter
ACM SIGMOD Record
http://www.cs.sfu.ca/CourseCentral/459/han/papers/chaudhuri97.pdf

Kimball, Ralph (2013)– Kimball Dimensional Modelling Techniques, http://www.kimballgroup.com/wp-content/uploads/2013/08/2013.09-Kimball-Dimensional-Modeling-Techniques11.pdf

Vaishnavi, Vijay; Kuechler, Bill; Petter, Stacie; -DESIGN SCIENCE RESEARCH IN INFORMATION SYSTEMS, http://desrist.org/desrist/content/design-science-research-in-information-systems.pdf